Schedule-free optimization of the transformers-based time series forecasting model

Kyrylo Yemets[1], Michal Gregus[2]
[1]Department of Artificial Intelligence, Lviv Polytechnic National University, Lviv, Ukraine
[2]Faculty of Management, Comenius University Bratislava, Bratislava, Slovak Republic

Article Info

Article history:

Received Jul 27, 2024
Revised Oct 26, 2024
Accepted Nov 14, 2024

Keywords:

AdaGrad
Forecasting
RMSprop
Schedule-Free AdamW
Schedule-Free SGD
Time series
Transformers

ABSTRACT

The task of time series forecasting is important for many scientific, technical, and applied fields, such as finance, economics, meteorology, medicine, transportation, and telecommunications. Existing methods, such as autoregressive models and moving average models, have their limitations, especially when working with non-stationary and seasonal data. In this work, the basic architecture of transformers was modified to solve time series forecasting problems. Additionally, state-of-the-art optimizers were investigated and experimentally compared, including AdamW, stochastic gradient descent (SGD), and new methods such as schedule-free SGD and schedule-free AdamW, to improve forecasting accuracy and the efficiency of the training procedure for the transformer architecture. Modeling was conducted on meteorological data that included seasonal time series. The accuracy evaluation of the optimization methods studied in this work was performed using a range of different performance indicators. The results showed that the new optimization methods significantly improve forecasting accuracy compared to the use of traditional optimizers.

Corresponding Author:

Kyrylo Yemets
Department of Artificial Intelligence, Lviv Polytechnic National University
Kniazia Romana str., 5, Lviv 79905, Ukraine
Email: kyrylo.v.yemets@lpnu.ua

1. INTRODUCTION

Time series forecasting is a relevant and extremely important task in various scientific, technical, and applied fields [1]. This task arises in the context of disciplines such as finance, economics, meteorology, medicine, transportation, telecommunications, and many others, where predicting future values of specific variables based on their historical data is necessary [2], [3]. The accuracy of solving the time series forecasting problem plays an important role and has a direct impact on the efficiency and economic benefit of decisions made. Particularly in the field of meteorology, the accuracy of weather forecasts is critically important for predicting natural disasters and planning activities in various sectors of the economy [4]. This task is an integral part of effective resource management and strategic planning.

Time series forecasting is a complex task due to the large number of factors affecting data variability and their interrelationships [5]. In particular, seasonal fluctuations, trends, random deviations, and other components can significantly complicate the forecasting process [6]–[8]. One of the important tasks is to adequately account for all these factors when building a time series forecasting model [9], [10]. Imperfection of data, the presence of noise, and gaps in the data can also reduce the accuracy of forecasts [11], [12]. In addition, the speed of changes in the environment or market may require constant updating of models and their adaptation to new conditions [13], [14].

As of today, there are many methods for time series forecasting [15]. Among them, traditional statistical methods should be highlighted, such as autoregressive models [16], moving average models [17], and their combinations, in particular autoregressive integrated moving average (ARIMA) [18], and other artificial neural networks (ANNs) [19]–[22]. Specifically, the latter method assumes a linear relationship between past and future values. However, weather patterns often exhibit non-linear behaviors due to the complex interactions between various atmospheric variables, which ARIMA models cannot capture effectively [23]. In addition, ARIMA models can become overly complex and computationally intensive when dealing with multiple seasonal patterns, which are common in weather data. In general, these methods work well for forecasting stationary processes, but weather data often exhibit seasonality and trends, requiring extensive pre-processing to achieve stationarity, which can be challenging and sometimes inadequate [24]. That's why these methods impose a number of limitations when forecasting complex, non-stationary time series.

The rapid development of machine learning technologies has led to the emergence of new, more powerful time series forecasting methods, such as neural networks [13], gradient boosting [25], and ensemble methods [26], [27]. In particular, recurrent neural networks (RNN) and long short-term memory (LSTM) have become popular for time series forecasting due to their ability to account for dependencies over large time intervals. However, they have a number of drawbacks. Specifically, RNNs often face problems of vanishing or exploding gradients during training, which complicates the training of models on long time series [28]. This leads to the model potentially losing information about previous states or becoming unstable. The effectiveness of LSTM strongly depends on the choice of hyperparameters (e.g., number of layers, memory cell size, and learning rates). Incorrect choice of hyperparameters can lead to poor model performance [29]. That's why, recently, transformers have become widely used-models that were initially developed for natural language processing [30]. They allow accounting for relationships between data at different time scales and provide high forecast accuracy.

The emergence of ANN for deep learning based on the transformer architecture opens up many new possibilities for solving various applied problems. Although the basic architecture of such ANNs is designed for natural language processing [31], there are many other tasks where this architecture can demonstrate significant advantages when applied, one of which is time series prediction. This gives rise to the first task of this study-modifying the basic architecture of transformers [31] to be able to solve time series forecasting problems. This is especially useful when we have a huge amount of cyclical data, such as in meteorology. Taking into account the volume of transformer-based forecasting models, it is important not only to have the correct model architecture but also to use the best optimizer during the training of such a model [32], [33]. The model architecture plays a significant role as it determines how information is processed and represented. However, even the best architecture may not achieve the expected results without proper tuning and optimization. An optimizer is an algorithm used to adjust the weights of a neural network to minimize the loss function [32]. The choice of optimizer can have a big impact on convergence speed, training stability, and final model accuracy.

There are many different optimizers, each with its own advantages and disadvantages. For example, stochastic gradient descent (SGD) is one of the simplest and most frequently used optimizers [34]. It is effective for large datasets but can suffer from slow convergence and get stuck in local minima. Adaptive methods, such as Adam, root mean square propagation (RMSprop), and adaptive gradient algorithm (AdaGrad), use different approaches to adjust the learning rate of each parameter, which can lead to faster and more stable convergence. In particular, RMSprop maintains an adaptive learning rate that changes for each parameter based on the average square of previous gradients [35]. This helps avoid the problem of large or small learning rates, which can be useful for models with high variability in parameter scale. RMSprop has several drawbacks that can affect optimization efficiency. One of the main problems is the bias in moment estimates due to the use of exponential smoothing to calculate the average square of gradients. This bias is particularly noticeable in the early stages of training and can negatively affect the initial phase of optimization. Additionally, RMSprop can be sensitive to initial conditions, meaning incorrect weight initialization can lead to convergence problems and getting stuck in local minima. Another important issue is the dependence on batch size. An inappropriate batch size can affect the calculation of the average square of gradients and overall optimization efficiency. AdaGrad modifies the learning rate based on the frequency of parameter updates, allowing rarely updated parameters to have a higher learning rate [36]. However, over time, the learning rate can decrease to very small values, which can slow down learning. Adaptive moment estimation (Adam) is one of the most popular optimizers that combines the advantages of both AdaGrad and RMSprop. It uses both the first moment (mean of gradients) and the second moment (mean of squared gradients) to dynamically adjust the learning rate [37]. This allows faster reaching of the loss function minimum, especially in the early stages of training. The Adam optimizer also has its drawbacks that should be considered. Similar to RMSprop, Adam uses exponential smoothing to calculate the first and second moments of gradients, which can lead to biased estimates in the early stages of training and affect optimization stability. Although Adam works well on many tasks, it may show poor generalization on test data. This is because adaptive methods can overfit to training data, especially if it contains noise. Adam's excessive adaptivity can lead to unstable convergence and getting stuck in local minima, which is especially problematic for complex or very noisy tasks. Additionally, Adam requires more computational resources compared to simpler methods like SGD, which can be a problem for large models or large datasets.

In general, the choice of optimizer depends on the specific task, model architecture, and dataset [30], [38]. An incorrect choice can lead to prolonged training, instability, or unsatisfactory results. As a result, success in time series forecasting depends not only on the model architecture but also on the effective choice of optimizer, which underscores the importance of its use when creating highly effective forecasting models based on artificial neural networks. Therefore, the second task of this study arises, which involves the correct selection of an optimizer for the modified transformer model that would provide the best model performance characteristics according to user-selected criteria.

That is why this work aims to modify the basic architecture of transformers to solve time series forecasting problems and to select, study, and experimentally analyze the latest optimization methods to improve the efficiency of the modified time series forecasting model in meteorology. The main contributions of this paper are the following:

- We modified the architecture of the transformer model by removing the tokenizer and embedding layer, as well as replacing positional encoding with sinusoidal positional encoding and batch normalization with layer normalization, which enabled effective solving of time series forecasting tasks in the case of analyzing a large amount of data with pronounced seasonality.
- We selected, studied, and conducted an experimental comparison of a number of state-of-the-art optimizers, especially their schedule-free versions, to improve forecasting accuracy and reduce the size of the modified transformer model and, accordingly, its training time when solving seasonal time series forecasting problems in the field of meteorology.

The paper is structured as follows: section 2 presents the modification of the transformer model for solving the time series forecasting problem, its architecture is presented, all changes made and their advantages are explained. This section also describes the principles of operation, advantages, and disadvantages of a number of state-of-the-art optimizers that were used in practical research to improve the efficiency of using the modified transformer model. Section 3 provides a description of the dataset used for modeling, indicators of the effectiveness of using the transformer model. Also, here, modeling of the work of a number of studied optimizers is carried out and the results of their work are summarized based on several criteria. A comparison of their work is performed and the choice of the best of them for practical implementation and use of the modified transformer model for forecasting seasonal time series in the field of meteorology is justified. Section 4 presents conclusions on the obtained results and describes prospects for further research.

2. METHODS

This section presents a modification of the transformer model for solving the time series forecasting problem, as well as the principles of operation, advantages, and disadvantages of a number of state-of-the-art optimizers for this model.

2.1. Modified architecture of the transformers for solving time series forecasting tasks

The transformer architecture has become one of the most popular methods in machine learning, especially in natural language processing [31]. Transformers are used in tasks such as text translation, question answering, and sentiment analysis, due to their ability to efficiently process long sequences of data and capture dependencies between distant elements. The main components of a transformer are the encoder and decoder, which allow the model to understand the context and structure of input data [31].

Transformers can also be effectively applied to time series forecasting. This is especially useful when analyzing many time series related to a single topic, such as weather conditions. In the case of weather forecasting, we have various variables such as temperature, humidity, atmospheric pressure, and precipitation, which interact with each other and have complex time dependencies. Transformers allow the model to capture these dependencies and use them for accurate prediction of future values.

The structure and main components of the modified transformer architecture for the time series forecasting task, including the encoder and decoder, are presented in Figure 1. It's worth noting a number of differences from the classic transformer architecture [31]:

- Removal of the tokenizer. In the classic transformer architecture, the tokenizer is used to convert input data into tokens. In the adapted architecture, this component was removed, simplifying the model and reducing its computational complexity.

- Removal of the embedding layer. In the classic architecture, the embedding layer is used to convert tokens into fixed-dimension vectors. In the adapted architecture, this layer was also removed, further simplifying the model.
- Sinusoidal positional encoding. Instead of traditional positional encoding, sinusoidal positional encoding was used. This encoding has no learnable parameters and is specifically designed to better reflect the characteristics of time sequences. It naturally gives more weight to recent elements and allows the model to preserve the order of data in the time sequence, which is critically important when analyzing time series.
- Layer normalization. In the classic architecture, batch normalization is often used, which normalizes the outputs of previous layers using statistics from the entire batch of data. However, for models working with time series, this approach may not be the best choice. Using layer normalization in the adapted architecture ensures training stability, as the normalization of outputs is carried out exclusively based on the distribution moments of a single layer and does not depend on the batch size. This mitigates problems associated with accounting for possible seasonal or cyclical components of time series.

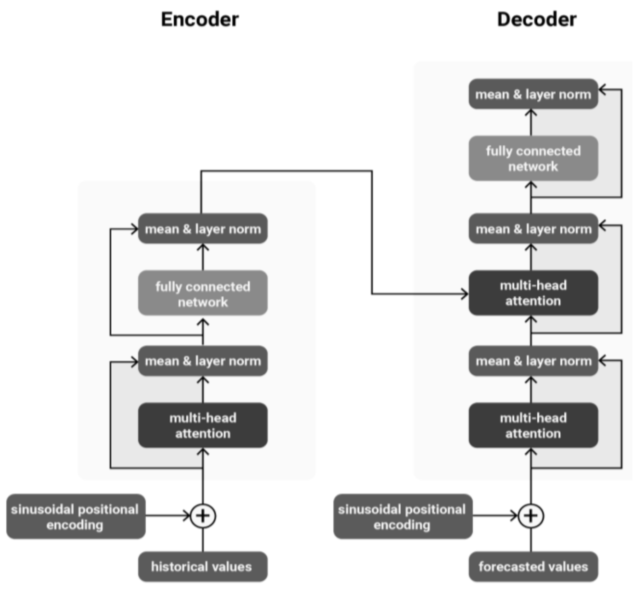

Figure 1. The modified architecture of the transformer model for time series forecasting

Removing the tokenizer and embedding layer, which are typically used for natural language processing, allows simplifying the model and reducing its computational complexity. Using sinusoidal positional encoding in the adapted architecture for time series forecasting provides better representation of the data sequence and preservation of their order. Layer normalization, in turn, provides greater training stability, regardless of the batch size, which is especially important when working with time series. All these changes make the adapted transformer architecture more adaptive and robust for analyzing various dynamic properties of data, improving overall accuracy and computational efficiency. This allows obtaining more accurate forecasts and reducing computational costs, which is important when solving time series forecasting problems.

2.2. Classical and state-of-the-arts optimizers for artificial neural networks

An important component of a highly effective time series forecasting model based on transformers is the optimizer it uses. In this paper, the effectiveness of several classical and completely new [31] optimization methods developed in 2024 was investigated. One of the classical optimizers used during the training of artificial neural networks is the SGD method. This method involves updating the model parameters based on the gradients of the loss function concerning the model parameters [34]. In the SGD method, parameter updates are carried out for each individual sample (or small batch of samples) from the training dataset. The update formula in this case looks like this [34]:

$$\theta_{t+1} = \theta_t - \eta \times grad(L_i(\theta_t)) \tag{1}$$

The SGD method has several important advantages [34]. Firstly, it is characterized by high speed, as parameter updates are carried out after each sample or small batch of samples. This makes it significantly faster than classical gradient descent, especially for large datasets. Secondly, the method is memory efficient, as it requires less memory, processing small portions of data at a time. However, the method also has certain disadvantages [34]. For example, parameter updates can be very noisy, which can lead to significant fluctuations in the loss function. Additionally, choosing the correct learning rate is critically important. Too high or too low learning rate can lead to convergence problems, complicating the model training process.

A new version of this method called schedule-free SGD [39], which was developed in 2024, aims to remove these drawbacks. One of the main advantages of schedule-free SGD is that it eliminates the need to adjust the learning rate. In classical SGD, this parameter requires careful tuning and may require the use of additional methods, such as learning rate schedulers, to dynamically change the learning rate during the optimization process. In schedule-free SGD, this necessity disappears, simplifying the model tuning process and reducing preparation time. In particular, its implementation can be described as follows [39]:

$$y_t = (1-\beta)z_t + \beta x_t \tag{2}$$

$$z_{t+1} = z_t - \gamma \times grad(f(y_t, \varsigma_t)) \tag{3}$$

$$x_{t+1} = (1-c_{t+1})x_t + c_{t+1}z_{t+1} \tag{4}$$

One of the widely used optimization methods is adaptive moment estimation with weight decay (AdamW), which is an improved version of the popular Adam optimizer, developed to solve problems related to weight regularization in neural networks [38]. The main idea of AdamW is to separate the process of weight updating and regularization, which allows avoiding some of the drawbacks of the original Adam algorithm [40]. Like Adam, AdamW combines ideas from AdaGrad and RMSprop methods for adaptive learning rate adjustment for each parameter. However, unlike Adam, AdamW applies weight regularization separately from gradient updates, providing better convergence and efficiency.

At each iteration, the AdamW optimizer calculates the gradient to minimize the function. The pseudocode for AdamW is shown in Figure 2. Then we update the weights taking into account weight regularization [39]:

$$\theta_t \leftarrow \theta_{t-1} - \gamma\lambda\theta_{t-1} \tag{5}$$

After that updating momentum of first and second order [39]:

$$m_t \leftarrow \beta_1 m_{t-1} + (1-\beta_1)g_t \tag{6}$$

$$v_t \leftarrow \beta_2 v_{t-1} + (1-\beta_2)g_t^2 \tag{7}$$

After this, bias-corrected momentums:

$$m_t \leftarrow m_t/(1-\beta_1^t) \tag{8}$$

$$\widehat{v_t} \leftarrow v_t/(1-\beta_2^t) \tag{9}$$

And we update the weights again considering the momentums:

$$\theta_t \leftarrow \theta_t - \gamma\widehat{m_t}/(\sqrt{\widehat{v_t}} + \epsilon) \tag{10}$$

AdamW has several important advantages. Firstly, thanks to the separation of weight updates and regularization, more stable and fast learning is ensured, contributing to better convergence. Secondly, effective weight regularization occurs separately, which allows avoiding problems associated with excessive weight regularization. Thirdly, like Adam, AdamW automatically adjusts the learning rate for each parameter, making it effective for different types of tasks. However, AdamW also has disadvantages. For example, proper tuning of hyperparameters can be difficult, as with other adaptive optimizers. Additionally, AdamW requires more computational resources compared to simpler methods such as SGD.

This problem of AdamW, such as hyperparameter tuning, is solved by its schedule-free modification, and this is how the modified schedule-free version of AdamW looks. In this version of the optimizer, the calculation of the second-order moment is replaced by a combination of interpolations and averages. The added version looks like this [39]:

$$\gamma_t = \gamma \times min(1, t/T_{warmup}) \tag{11}$$

$$z_{t+1} = z_t - \gamma_t g_t / (\sqrt{\hat{v}_t} + \epsilon) - \gamma_t \lambda y_t \quad (12)$$

$$c_{t+1} = \frac{\gamma_t^2}{\sum_{i=1}^{t} \gamma_i^2} \quad (13)$$

$$x_{t+1} = (1 - c_{t+1}) x_t + c_{t+1} z_{t+1} \quad (14)$$

These optimizers described above were used in this work to conduct experimental studies to determine the best of them when solving the time series forecasting problem based on the modified transformer model.

3. RESULTS AND DISCUSSION

The transformer model for time series forecasting with different optimizers was trained on a weather dataset [41]. This dataset contains 3010 daily time series representing changes in four weather variables: rain, minimum temperature, maximum temperature, and solar radiation, measured at meteorological stations in Australia. This dataset, provided by the Australian Bureau of Meteorology (BoM), includes current weather data for specific stations, daily forecasts for all Australian forecast locations, agricultural bulletins with summarized weather observations for each state or territory, and satellite images in GeoTIFF format. The dataset is available in XML and JSON formats and can be accessed through an anonymous FTP server. The dataset is well-structured and offers features for automatic retrieval and parsing of data into ordered dataframes. The data has applications in agriculture, mapping renewable energy potential, and planning for municipalities regarding extreme weather events and infrastructure needs.

Experimental studies on the effectiveness of different optimizers were performed by running the model and the corresponding optimizer with different context lengths. It was a multiple of the seasonality of the data in the dataset, which essentially represents one month. Thus, the transformer model was trained on lengths of 30, 60, and 90 days. The evaluation of the model's performance was based on the following indicators [27], [34]:

– Mean absolute error (MAE):

$$MAE = \frac{1}{n} \sum_{i=1}^{n} |actual_i - forecast_i| \quad (15)$$

– Mean square error (MSE):

$$MSE = \frac{1}{n} \sum_{i=1}^{n} (forecast_i - actual_i)^2 \quad (16)$$

– Root mean square error (RMSE):

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (forecast_i - actual_i)^2} \quad (17)$$

– Symmetric mean absolute percentage error (SMAPE):

$$SMAPE = \frac{1}{n} \sum_{i=1}^{n} \frac{|forecast_i - actual_i|}{(|forecast_i| + |actual_i|)/2} \quad (18)$$

– Mean absolute scaled error (MASE):

$$MASE = \frac{1}{n} \sum_{i=1}^{n} \frac{|forecast_i - actual_i|}{|actual_i - previous_i|} \quad (19)$$

We also used the number of trainable model parameters, expressed in thousands, which can show how many resources we will need for its training and what the delay will be when executed in a production environment. The results of the experiments based on all the above performance indicators are presented in Table 1. The obtained results show that the SGD optimizer gives the worst results. For example, for a context length of 60 days, the MASE value is 4.34, which is significantly worse compared to other optimizers. Using the schedule-free SGD optimizer improves these indicators, reducing MASE to 1.116 for the same context length, which is an improvement of approximately 74%. The AdamW optimizer shows better results compared to schedule-free SGD, with a MASE value of 0.954 for a 60-day context length, which is an improvement of 14%. However, the best results are demonstrated by the schedule-free AdamW optimizer, where for a 60-day context length, MASE is 0.987, which is slightly worse than AdamW, but overall shows stable results for all context lengths. Thus, from the point of view of stability and overall efficiency, schedule-free AdamW is the best choice among all tested optimizers. However, if we consider the results in terms of SMAPE, the schedule-free SGD optimizer demonstrates the best performance. For example, for a 30-day context length, SMAPE is 0.651, which is better than all other optimizers. For 60-day and 90-day context lengths, SMAPE also remains lower than other optimizers, making schedule-free SGD the best choice for the SMAPE metric.

Table 1. Efficiency estimates of the transformer model in solving the time series forecasting problem using different optimizers studied in the work

Context length	MASE	SMAPE	MAE	MSE	RMSE	Model size
Schedule-free AdamW optimizer						
30	0.94	0.691	2.041	17.606	2.832	80359
60	0.987	0.691	2.154	18.781	2.963	82279
90	0.931	0.686	2.046	17.83	2.836	84199
AdamW optimizer						
30	1.206	0.666	2.295	19.248	3.056	80359
60	0.954	0.703	2.071	17.955	2.87	82279
90	1.063	0.7	2.307	20.074	3.148	84199
Schedule-free SGD optimizer						
30	1.122	0.651	2.556	20.486	3.203	80359
60	1.116	0.658	2.574	20.222	3.238	82279
90	1.184	0.652	2.621	20.669	3.296	84199
SGD optimizer						
30	1.236	0.724	2.655	21.834	3.489	80359
60	4.34	1.378	7.704	107.351	8.632	82279
90	1.573	0.703	2.792	23.33	3.593	84199

Figure 2 shows a comparative graph of optimizers with different context sizes, relative to two key metrics: MASE and SMAPE. The size of each point corresponds to the number of model parameters. The closer the point is to the origin of the coordinate system, the better it is suited for model training. As can be seen from the graph, the schedule-free versions of SGD and AdamW performed the best. Depending on which metric is prioritized, there will be different results. For MASE, it's schedule-free AdamW, and for SMAPE, it's schedule-free SGD. In the future, we can investigate how schedule-free optimizers affect the confidence interval of time series model predictions.

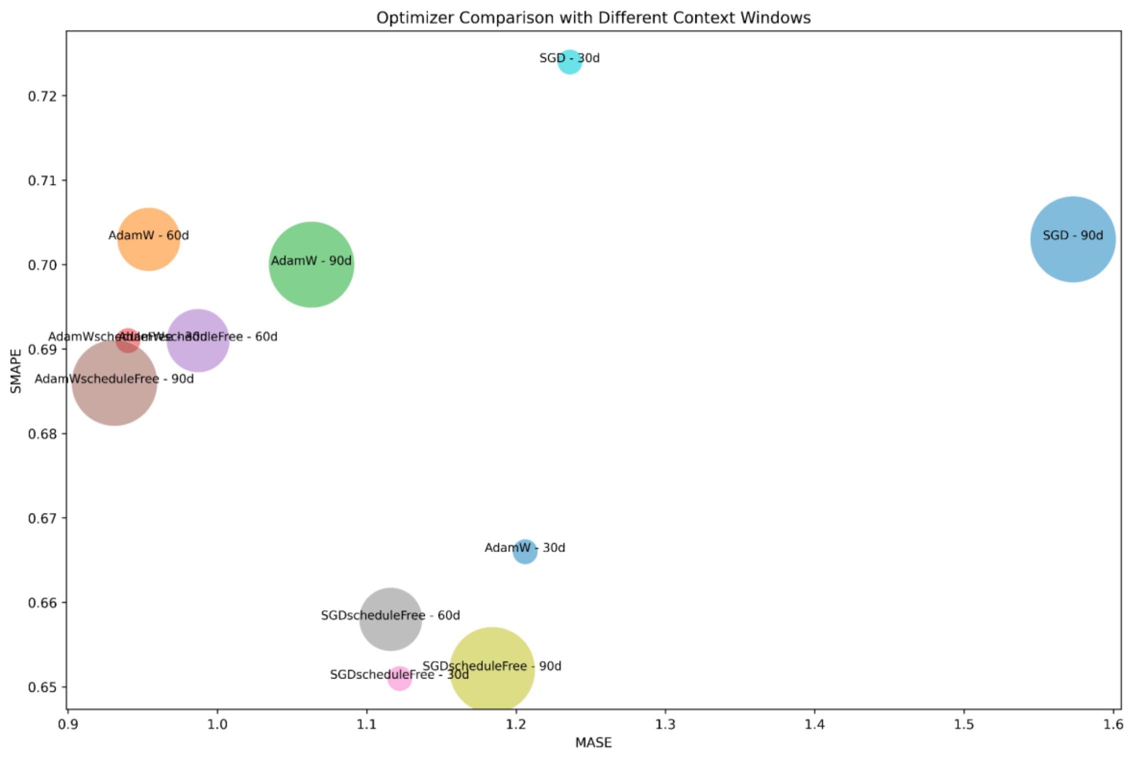

Figure 2. Graph with comparison of optimizers by SMAPE and MASE metrics

Future research will address several key areas to enhance the accuracy and applicability of forecasting models. Firstly, while current studies have utilized a single dataset, future work will expand this research to include multiple datasets, allowing for more robust and generalizable findings. Additionally, the impact of confidence interval magnitude on forecasting accuracy has not yet been explored as a distinct parameter; investigating this aspect could provide valuable insights. Research will also focus on identifying the most effective optimizers for improving forecast performance. Another promising avenue for future research is the development of transformer-based ensemble models, which could significantly enhance forecast accuracy across various application areas [42].

4. CONCLUSION

Time series forecasting is extremely important in various scientific, technical, and applied fields such as finance, economics, meteorology, medicine, transportation, and telecommunications. This task helps predict future values of variables based on their historical data, which has a direct impact on the efficiency and economic benefit of decisions made. In meteorology, the accuracy of weather forecasts is critically important for predicting natural disasters and planning activities in various sectors of the economy. In this paper, the basic architecture of transformers was modified to solve time series forecasting problems. The removal of the tokenizer and embedding layer, as well as the replacement of positional encoding with sinusoidal positional encoding and batch normalization with layer normalization, provided the ability to work with data with pronounced seasonality. State-of-the-art optimizers, especially their schedule-free versions, were studied and experimentally compared to improve forecasting accuracy and reduce the size of the transformer model and its training time. Modeling was conducted by using the modified transformer architecture based on a large set of meteorological data, which includes various variables such as temperature, humidity, atmospheric pressure, and precipitation. The data was pre-processed to remove noise and gaps. The model was trained on historical data with different context window sizes and for each optimizer, after which testing was conducted on new data to evaluate the accuracy of forecasts. The results of different optimizers were compared based on criteria such as SMAPE, MASE, MAE, MSE, and RMSE. The results showed that the modified schedule-free optimizers significantly improved the accuracy of seasonal time series forecasting compared to classical methods. The most relevant are both schedule-free SGD and schedule-free AdamW. As a result of comparing optimizers, schedule-free AdamW proved to be the best. As can be seen from the results in the table, with this optimizer, the model shows the best metrics with the smallest context, which also makes the model smaller and thus faster both during training and inference.

ACKNOWLEDGEMENTS

The authors thank Dr. Ivan Izonin for his invaluable guidance and recommendations, which significantly enhanced the presentation of this research. Prof. Michal Gregus was supported by the Slovak Research and Development Agency under the contract No. APVV 19-0581. This work is funded by the European Union's Horizon Europe research and innovation program under grant agreement No 101138678, project ZEBAI (Innovative methodologies for the design of Zero-Emission and cost-effective Buildings enhanced by Artificial Intelligence).

REFERENCES

[1] M. Geurts, G. E. P. Box, and G. M. Jenkins, "Time series analysis: forecasting and control," *Journal of Marketing Research*, vol. 14, no. 2, May 1977, doi: 10.2307/3150485.

[2] R. Tkachenko, "An integral software solution of the sgtm neural-like structures implementation for solving different data mining tasks," in *Lecture Notes in Computational Intelligence and Decision Making*, vol. 77, 2022, pp. 696–713, doi: 10.1007/978-3-030-82014-5_48.

[3] M. Havryliuk, R. Kaminskyy, K. Yemets, and T. Lisovych, "Interactive information system for automated identification of operator personnel by schulte tables based on individual time series," in *Advances in Artificial Systems for Logistics Engineering III*, vol. 180, 2023, pp. 372–381, doi: 10.1007/978-3-031-36115-9_34.

[4] S. Raksha, J. S. Graceline, J. Anbarasi, M. Prasanna, and S. Kamaleshkumar, "Weather forecasting framework for time series data using intelligent learning models," in *2021 5th International Conference on Electrical, Electronics, Communication, Computer Technologies and Optimization Techniques (ICEECCOT)*, IEEE, Dec. 2021, pp. 783–787, doi: 10.1109/ICEECCOT52851.2021.9707971.

[5] W. Sulandari, S. Subanar, S. Suhartono, H. Utami, M. H. Lee, and P. C. Rodrigues, "SSA-based hybrid forecasting models and applications," *Bulletin of Electrical Engineering and Informatics*, vol. 9, no. 5, pp. 2178–2188, Oct. 2020, doi: 10.11591/eei.v9i5.1950.

[6] D. K. Singh and N. Rawat, "Machine learning for weather forecasting: xgboost vs svm vs random forest in predicting temperature for visakhapatnam," *International Journal of Intelligent Systems and Applications*, vol. 15, no. 5, pp. 57–69, Oct. 2023, doi: 10.5815/ijisa.2023.05.05.

[7] D. Uhryn *et al.*, "Modelling of an intelligent geographic information system for population migration forecasting," *International Journal of Modern Education and Computer Science*, vol. 15, no. 4, pp. 69–79, Aug. 2023, doi: 10.5815/ijmecs.2023.04.06.

[8] O. Korystin, S. Nataliia, and O. Mitina, "Risk forecasting of data confidentiality breach using linear regression algorithm," *International Journal of Computer Network and Information Security*, vol. 14, no. 4, pp. 1–13, Aug. 2022, doi: 10.5815/ijcnis.2022.04.01.

[9] H. Dalkani, M. Mojarad, and H. Arfaeinia, "Modelling electricity consumption forecasting using the markov process and hybrid features selection," *International Journal of Intelligent Systems and Applications*, vol. 13, no. 5, pp. 14–23, Oct. 2021, doi: 10.5815/ijisa.2021.05.02.

[10] P. B. Angon, I. Salehin, M. M. R. Khan, and S. Mondal, "Cropland mapping expansion for production forecast: rainfall, relative humidity and temperature estimation," *International Journal of Engineering and Manufacturing*, vol. 11, no. 5, pp. 25–40, Oct. 2021, doi: 10.5815/ijem.2021.05.03.

[11] O. Mulesa, F. Geche, A. Batyuk, and V. Buchok, "Development of combined information technology for time series prediction," in *Advances in Intelligent Systems and Computing*, 2018, pp. 361–373, doi: 10.1007/978-3-319-70581-1_26.

[12] M. R. A. Yudianto, T. Agustin, R. M. James, F. I. Rahma, A. Rahim, and E. Utami, "Rainfall forecasting to recommend crops varieties using moving average and naive bayes methods," *International Journal of Modern Education and Computer Science*, vol. 13, no. 3, pp. 23–33, Jun. 2021, doi: 10.5815/ijmecs.2021.03.03.

[13] A. Casolaro, V. Capone, G. Iannuzzo, and F. Camastra, "Deep learning for time series forecasting: advances and open problems," *Information*, vol. 14, no. 11, Nov. 2023, doi: 10.3390/info14110598.

[14] Kasliono, Suprapto, and F. Makhrus, "Point based forecasting model of vehicle queue with extreme learning machine method and correlation analysis," *International Journal of Intelligent Systems and Applications*, vol. 13, no. 3, pp. 11–22, Jun. 2021, doi: 10.5815/ijisa.2021.03.02.

[15] Z. Liu, Z. Zhu, J. Gao, and C. Xu, "Forecast methods for time series data: a survey," *IEEE Access*, vol. 9, pp. 91896–91912, 2021, doi: 10.1109/ACCESS.2021.3091162.

[16] H. Hermansah, D. Rosadi, A. Abdurakhman, and H. Utami, "Automatic time series forecasting using nonlinear autoregressive neural network model with exogenous input," *Bulletin of Electrical Engineering and Informatics*, vol. 10, no. 5, pp. 2836–2844, Oct. 2021, doi: 10.11591/eei.v10i5.2862.

[17] D. M. Khairina, R. Khairunnisa, H. R. Hatta, and S. Maharani, "Comparison of the trend moment and double moving average methods for forecasting the number of dengue hemorrhagic fever patients," *Bulletin of Electrical Engineering and Informatics*, vol. 10, no. 2, pp. 978–987, Apr. 2021, doi: 10.11591/eei.v10i2.2711.

[18] D. G. Taslim and I. M. Murwantara, "Comparative analysis of arima and lstm for predicting fluctuating time series data," *Bulletin of Electrical Engineering and Informatics*, vol. 13, no. 3, pp. 1943–1951, Jun. 2024, doi: 10.11591/eei.v13i3.6034.

[19] Z. Hu, Y. V Bodyanskiy, N. Y. Kulishova, and O. K. Tyshchenko, "A multidimensional extended neo-fuzzy neuron for facial expression recognition," *International Journal of Intelligent Systems and Applications*, vol. 9, no. 9, pp. 29–36, Sep. 2017, doi: 10.5815/ijisa.2017.09.04.

[20] Z. Hu, I. A. Tereykovski, L. O. Tereykovska, and V. V Pogorelov, "Determination of structural parameters of multilayer perceptron designed to estimate parameters of technical systems," *International Journal of Intelligent Systems and Applications*, vol. 9, no. 10, pp. 57–62, Oct. 2017, doi: 10.5815/ijisa.2017.10.07.

[21] Z. Hu, Y. Khokhlachova, V. Sydorenk, and I. Opirskyy, "Method for optimization of information security systems behavior under conditions of influences," *International Journal of Intelligent Systems and Applications*, vol. 9, no. 12, pp. 46–58, Dec. 2017, doi: 10.5815/ijisa.2017.12.05.

[22] D. Salinas, V. Flunkert, J. Gasthaus, and T. Januschowski, "DeepAR: probabilistic forecasting with autoregressive recurrent networks," *International Journal of Forecasting*, vol. 36, no. 3, pp. 1181–1191, Jul. 2020, doi: 10.1016/j.ijforecast.2019.07.001.

[23] V. I. Kontopoulou, A. D. Panagopoulos, I. Kakkos, and G. K. Matsopoulos, "A review of arima vs. machine learning approaches for time series forecasting in data driven networks," *Future Internet*, vol. 15, no. 8, Jul. 2023, doi: 10.3390/fi15080255.

[24] S. Chen, R. Lin, and W. Zeng, "Short-term load forecasting method based on ARIMA and LSTM," in *2022 IEEE 22nd International Conference on Communication Technology (ICCT)*, IEEE, Nov. 2022, pp. 1913–1917, doi: 10.1109/ICCT56141.2022.10073051.

[25] N. H. A. Malek, W. F. W. Yaacob, Y. B. Wah, S. A. Md Nasir, N. Shaadan, and S. W. Indratno, "Comparison of ensemble hybrid sampling with bagging and boosting machine learning approach for imbalanced data," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 29, no. 1, pp. 598–608, Jan. 2022, doi: 10.11591/ijeecs.v29.i1.pp598-608.

[26] L. A. Ortega, R. Cabañas, and A. R. Masegosa, "Diversity and generalization in neural network ensembles," in *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics*, 2022, pp. 11720–11743.

[27] A. Jadhav, S. K. Shandilya, I. Izonin, and R. Muzyka, "Multi-step dynamic ensemble selection to estimate software effort," *Applied Artificial Intelligence*, vol. 38, no. 1, Dec. 2024, doi: 10.1080/08839514.2024.2351718.

[28] N. Nikentari and H.-L. Wei, "Multi-task learning using non-linear autoregressive models and recurrent neural networks for tide level forecasting," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 14, no. 1, pp. 960–970, Feb. 2024, doi: 10.11591/ijece.v14i1.pp960-970.

[29] A. Muneer, R. F. Ali, A. Almaghthawi, S. M. Taib, A. Alghamdi, and E. A. A. Ghaleb, "Short term residential load forecasting using long short-term memory recurrent neural network," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 12, no. 5, pp. 5589–5599, Oct. 2022, doi: 10.11591/ijece.v12i5.pp5589-5599.

[30] A. Barlybayev and B. Matkarimov, "Development of system for generating questions, answers, distractors using transformers," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 14, no. 2, pp. 1851–1863, Apr. 2024, doi: 10.11591/ijece.v14i2.pp1851-1863.

[31] A. Vaswani *et al.*, "Attention is all you need," *Advances in Neural Information Processing Systems*, vol. 1, 2017.

[32] C. Ganguli, S. K. Shandilya, M. Nehrey, and M. Havryliuk, "Adaptive artificial bee colony algorithm for nature-inspired cyber defense," *Systems*, vol. 11, no. 1, Jan. 2023, doi: 10.3390/systems11010027.

[33] M. Havryliuk, N. Hovdysh, Y. Tolstyak, V. Chopyakb, and N. Kustra, "Investigation of pnn optimization methods to improve classification performance in transplantation medicine," in *CEUR Workshop Proceedings*, 2023, pp. 338–345.

[34] I. Izonin, R. Tkachenko, R. Holoven, K. Yemets, M. Havryliuk, and S. K. Shandilya, "SGD-based cascade scheme for higher degrees wiener polynomial approximation of large biomedical datasets," *Machine Learning and Knowledge Extraction*, vol. 4, no. 4, pp. 1088–1106, Nov. 2022, doi: 10.3390/make4040055.

[35] J. Huang, "RMSProp," *Cornell University*. Accessed: Sep. 13, 2024. [Online]. Available: https://optimization.cbe.cornell.edu/index.php?title=RMSProp

[36] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *Journal of Machine Learning Research*, vol. 12, no. 61, pp. 2121–2159, 2011.

[37] D. P. Kingma and J. Ba, "Adam: a method for stochastic optimization," *arXiv-Computer Science*, pp. 1-15, 2014.

[38] N. Xiao, X. Hu, X. Liu, and K.-C. Toh, "Adam-family methods for nonsmooth optimization with convergence guarantees," *arXiv-Mathematics*, pp. 1-53, 2023.

[39] A. Defazio, X. A. Yang, H. Mehta, K. Mishchenko, A. Khaled, and A. Cutkosky, "The road less scheduled," in *38th Conference on Neural Information Processing Systems (NeurIPS 2024)*, 2024.

[40] PyTorch, "AdamW: implements adamw algorithm," *PyTorch Contributors*. Accessed: Jul. 13, 2024. [Online]. Available: https://pytorch.org/docs/stable/generated/torch.optim.AdamW.html

[41] A. H Sparks, M. Padgham, H. Parsonage, and K. Pembleton, "Bomrang: fetch Australian government bureau of meteorology data in r," *The Journal of Open Source Software*, vol. 2, no. 17, Sep. 2017, doi: 10.21105/joss.00411.

[42] I. Izonin, R. Muzyka, R. Tkachenko, I. Dronyuk, K. Yemets, and S.-A. Mitoulis, "A method for reducing training time of ML-based cascade scheme for large-volume data analysis," *Sensors*, vol. 24, no. 15, Jul. 2024, doi: 10.3390/s24154762.

BIOGRAPHIES OF AUTHORS

Kyrylo Yemets is a Ph.D. student at the Department of Artificial Intelligence of Lviv Polytechnic National University, Ukraine, and a machine learning engineer. He received an M.Sc. degree in computer science from the National Technical University "Kharkiv Polytechnic Institute", Ukraine in 2021. His research interests include time series, natural language processing, transformers, and ensemble methods where he is the author/co-author of over 10 research publications. He can be contacted at email: kyrylo.v.yemets@lpnu.ua.

Prof. Michal Greguš is a Professor of the Faculty of Management, Comenius University Bratislava, Bratislava, Slovak Republic. He finished his university studies with summa cumlaude and obtained his Ph.D. degree in the field of mathematical analysis at the Faculty of Mathematics and Physics at Comenius University in Bratislava. He has been working previously in the field of functional analysis and its applications. At present, his research interests are in management information systems, in modelling of economic processes and in business analytics. He can be contacted at email: michal.gregus@fm.uniba.sk.